

# Weekly Report (2017.12.25-2017.12.31)

TASK	DEADLINE	CURRENT PROGRESS
可视交互引擎论文	2017.12月底	目前进展：补充了沉浸式可视分析架构与平台 存在问题：临近期末，时间不够充裕 计划：提高效率，多读论文
AI课程论文	2018.1.10	目前进展：确定主题，查阅了一些相关论文 计划：阅读相关论文，确定大纲，准备报告PPT，6号做 presentation，撰写综述报告
游戏设计与开发课程项目	2018.1.15	目前进展：确定项目内容，没有开始做 计划：熟悉 WebGL 基本语法，着手开始做
VR课程项目	2018.1.23	目前进展：安装学习了Unity with IOS 计划：学习 Unity with IOS 的3D图形渲染，实际进行开发实验

## Done

### 1. Paper writing.

Add content to the subsection that Methods and Techniques of intelligent visual analysis. Due to the lack of time, the addition is not too much.

### 2. Paper reading.

Bubble Treemaps for Uncertainty Visualization.

This work proposes a layout based on circle packing to use space purposely, achieving a reasonable trade-off between a compact representation of the hierarchy and its inherent information, and defines node contours analytically, resulting in a new parameter domain to be used for additional visual variables, in particular, for uncertainty visualization.

### 3. AI survey.

Look up to the relevant contents of the alternative topics. And determine the topic is Text Categorization. Here are some papers to read in detail:

<http://www.ijcaonline.org/archives/volume166/number11/saxena-2017-ijca-914145.pdf>

<http://journals.sagepub.com/doi/abs/10.1177/0165551514558172>

<http://www.ijcsec.com/2017/5316121618.pdf>

[https://link.springer.com/chapter/10.1007/978-81-322-1602-5\\_75](https://link.springer.com/chapter/10.1007/978-81-322-1602-5_75)

<http://www.ijcaonline.org/archives/volume166/number11/saxena-2017-ijca-914145.pdf>

### 4. CG homework.

Due to my negligence, I did the wrong homework of CG last week. So I did it again. Reimplement the Interval Scanline Z-buffer algorithm. This algorithm is more efficient than the scanline Z-buffer algorithm which is done last week. This algorithm reduces a number of visibility judgments, which is the application of the theory that interval is consistent.

The report is:

# 区间扫描线Z-buffer算法程序说明

姓名：魏雅婷 班级：博士1班 学号：11721016

## 1. 编程环境

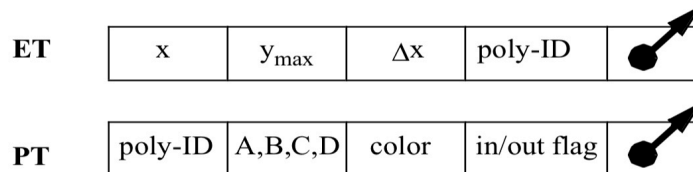
VS2013+OpenGL

## 2. 用户界面

点击运行后，程序加载预先准备好的 flower.obj 文件，应用区间扫描线Z-buffer算法预处理 obj 数据，并使用 OpenGL 在 win32 控制台上绘制出 3D 模型，并显示 3D 模型的面片数、顶点数以及算法运行时间等一系列统计信息。程序截图见本说明的运行结果展示小节。

## 3. 数据结构

本项目中区间扫描线 Z-buffer 算法的实现完全参照老师上课的ppt中所讲述的算法实现流程，以下详细介绍算法相关的数据结构。



**ET**: 边链表,根据ymax(边的上端点y坐标)将

边放入相应的类中

x:边的上端点的x坐标

dx:相邻两条扫描线交点的x坐标差dx (-1/k)

dy:边跨越的扫描线数目

id:边所属多边形的编号

**PT**: 多边形链表

a,b,c,d:多边形所在平面的方程系数

$ax+by+cz+d=0$

id:多边形的编号

dy:多边形跨越的扫描线数目

color:多边形的颜色

flag:记录扫描线是否进入该多边形

算法中间的数据结构: AET (活化边链表), IPL (活化多边形链表)

ppt中伪代码截图:

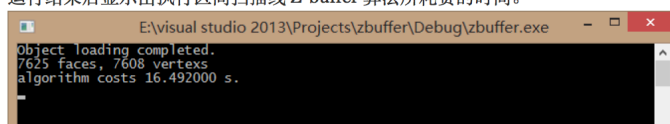
```
build ET, PT          -- all polys+BG poly
AET := IPL := Nil;
for y := ymin to ymax do
    e1 := first_item ( AET ); IPL := BG;
    while (e1.x <> MaxX) do
        e2 := next_item (AET);
        poly := closest poly in IPL at [(e1.x+e2.x)/2, y]
        draw_line(e1.x, e2.x, poly-color);
        update IPL (flags); e1 := e2;
    end-while;
    IPL := NIL; update AET;
end-for;
```

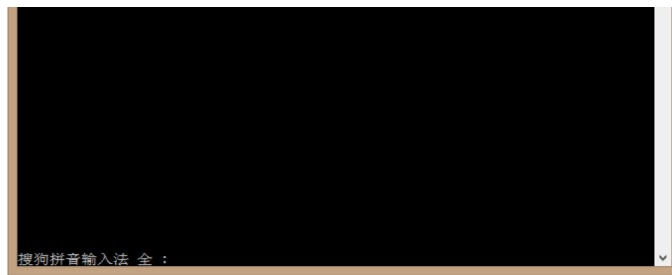
## 4. 加速说明

相比扫描线算法，区间扫描线算法利用区间的连贯性，将每一条扫描线分割成几个区间，并且对每一个区间只做一次可见性判断，省去了大量的可见性判断，提高效率。

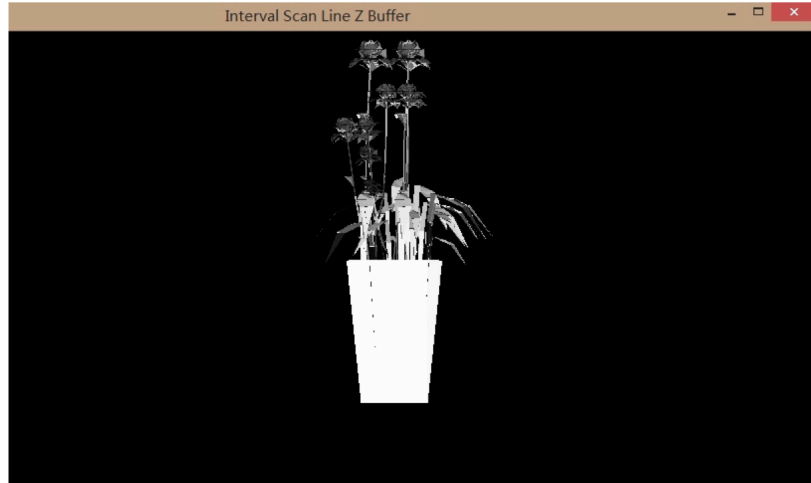
## 5. 运行结果展示

a. Object 文件加载完毕后，程序显示 Object 文件的相关统计信息，包括 3D 模型的面片数以及顶点数，并在程序运行结束后显示出执行区间扫描线 Z-buffer 算法所耗费的时间。





b. 程序渲染出的 3D 模型结果图。obj 文件中原本没有颜色值，结果图中的颜色是根据像素点的相对深度进行确定的。



## 6. 参考资料

a. 老师ppt算法与实现流程

## 7. 总结与感悟

一开始没有认真审题，忽略了区间两个字，做成了扫描线 Z-buffer 的实现，后来经同学提醒才发现，但是我觉得把两个算法都实现一遍的结果就是自己对两个算法更进一步的理解。按照老师ppt上讲的来说的话，应该是，区间扫描线算法比扫描线算法的效率更高一些，但是，我做出的结果却是，区间扫描线算法的运行效率比扫描线的低很多，一开始一度认为自己的区间扫描线算法实现有问题，对着 ppt review 了好几遍，并没有发现什么问题。于是就开始反思，发现问题所在，我实现的扫描线算法中省去了一个活化多边形链表（因为在ppt中讲解的算法实现过程中，活化多边形链表并没有非常必要，于是就省去了），这就意味着省去了一个链表的遍历开销，而在区间扫描线算法中，活化多边形链表的遍历是非常必要，而且开销很大，所以，我觉得虽然区间扫描线算法省去了一些可见性判断，但是其他开销也很大。

## TODO

1. Write paper.
2. Prepare for the group meeting report.
3. Prepare for the AI report PPT and write survey.
4. Set out to the Game design and development course project.